ARCHITECTURE OVERVIEW, OCTOBER 2026

GPAP – Global Pathogen Analysis Platform



01 Objectives

What the platform will do

Fast sample navigation

Filter by country, species or resistance genes..

Modular tools

Notebook style arbitrary bioinformatics tools execution

Custom pipelines

Easiness defining custom pipelines with different tools

Sharing/collaboration

Work together in projects across organizations, sharing samples, metadata, results

AI Integration

Agentic AI together with traditional ml algorithms

Provenance

Standardization of results and executions

02 STRUCTURE

How the code is organized (modular monolith)

Identity
Who is logged in (Keycloak)

Organizations
Members, roles and invitations

Upload and storage
Files, deduplication

Samples
Metadata and attached files

Execution engine
Tools, pipelines definitions, runs

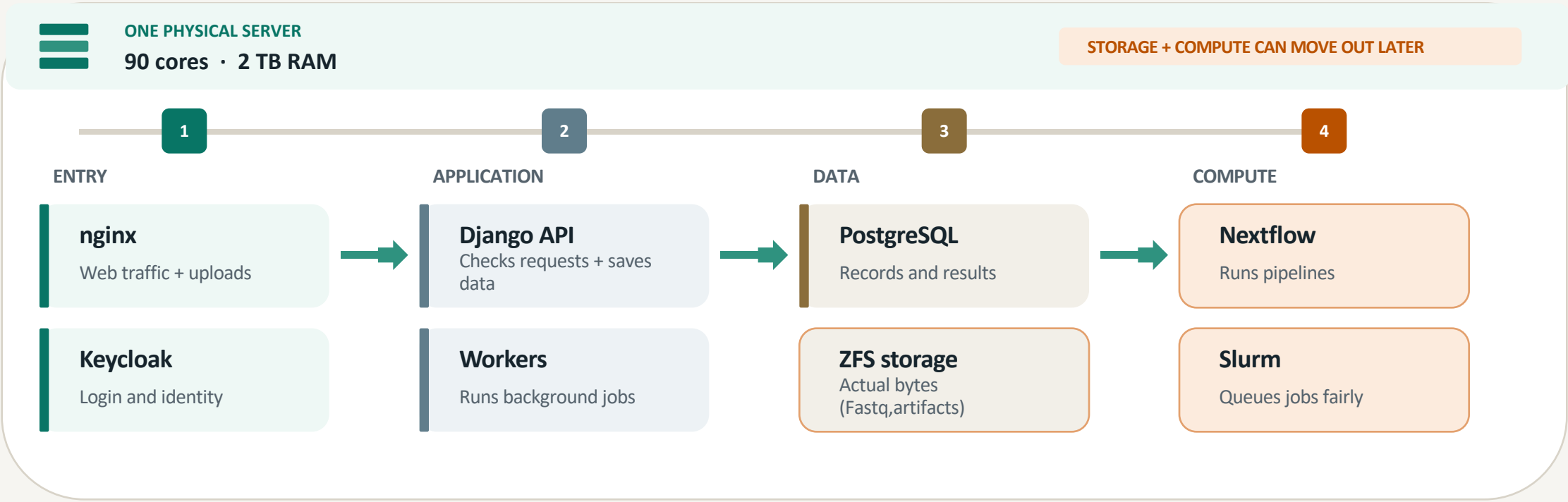
Results
Tool families denormalized tables

Sharing
Projects, access levels, folders

Events
Notifications, audit log, background work

Module independence – connection via api interfaces

What runs on the server today



TODAY Simple to operate on one server; storage and compute are the cleanest boundaries to separate later.

Sample – Main logical unit

Sample

Belongs to one organization
Has an id that cannot be guessed
Keeps a version, so edits never clash

Files

FASTQ, FASTA or nanopore reads. Each file is stored only once.

Metadata

Date, country, host, lab, and an extra row for standard fields.

Results

We keep the tool’s original output and copy the key values into tables.

Deduplication on blobs. Storage is hidden behind an interface. Easy integration with different infrastructures

Different sample rows can point to the same bytes

ACTUAL SAMPLE RECORDS

samples.sample

id	public_id	owner_org	name	country	collected	version	extra (jsonb)
50230	01J9ZK…a1	RKI	EC-000	DE	2025-03-11	3	{"host_age": 54}
50231	01J9ZK…b7	RKI	EC-001	DE	2025-03-14	1	{}
50232	01J9ZM…04	SSI	DK-114	DK	2025-03-15	2	{"farm": "F-12"}

SAMPLE_FILES — ONE LINK PER SAMPLE

sample_id	file role	blob_id
50230	reads	B-009
50231	reads	B-009
50232	reads	B-009

same SHA-256

BLOBS — ONE ROW PER UNIQUE HASH

blob_id	sha256	size	path
B-009	a91c…e42d	4.8 GB	a9/1c/…

ONE PHYSICAL OBJECT · blobs/a9/1c/a91c…e42d.fastq

SHA-256 identifies content: new samples add link rows, while identical bytes are stored only once.

06 SPEED

How search stays fast

Detailed tables
Samples, files, results

Updates
Sent whenever data changes

Search table
One row per sample

Results
Back in milliseconds

Indexes
Only for the filters people actually use

Busy tables
Old entries are cleaned up regularly

Dashboards
Counts are prepared in advance

Long lists
Loaded 50 at a time

Why metadata matters

1

A few core fields

Country, date, host, source and lab. These are the ones people filter on.

2

One extra row per sample

Everything else, following standards like MIxS or PHA4GE. We can add fields without changing the database.

3

Import and export, later

Reading from and submitting to ENA and NCBI, in their TSV and XML formats.

Good metadata are essential - Focus in helping users in their submission

Tools – Easy to integrate (like a black box)

Description
What goes in, what comes out, which settings

Container
Docker or Apptainer, fixed to one exact version

Original output
Kept as it is, for good

Parser
Pulls the important values into tables

Every result remembers
Which tool and version, which input files, which settings, and which reference database was used.

One Nextflow batch creates one run per sample

INITIATE RUN

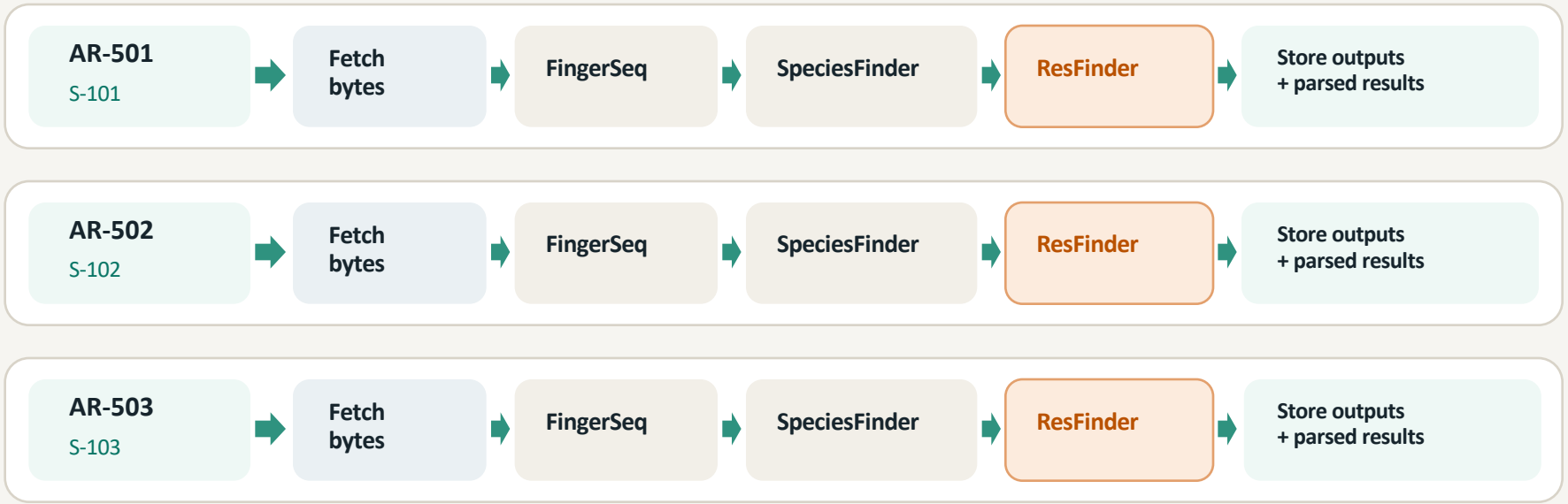
WorkflowBatch

WB-042

3
samples

Owns orchestration, shared parameters and overall status.

1 sample = batch of 1



OBJECT MODEL 1 WorkflowBatch coordinates N AnalysisRuns. Scaling computing engine essential (slurm , scheduling etc)

10 SHARING

How sharing works

Organization

Owns its samples. Each user is in one organization and also has a test sandbox.

Project

A shared space with viewers, editors and admins. Everyone in it sees the same samples.

Folder

Your own bookmarks. Folders never change who can see what.

What others see	Metadata	Results	Files
Metadata only	Yes	No	No
Metadata and results	Yes	Yes	No
Full	Yes	Yes	Yes

Samples are linked into projects, not copied. If the owner removes one, nothing is deleted.

11 PERMISSIONS

How we check permissions

Request
Login says who you are

Your context
Your organization and your project roles

Access check
Can you do this, on this sample?

Answer
Yes, or not found

Viewer
Can look at samples and analyses

Editor
Can also run tools and download results

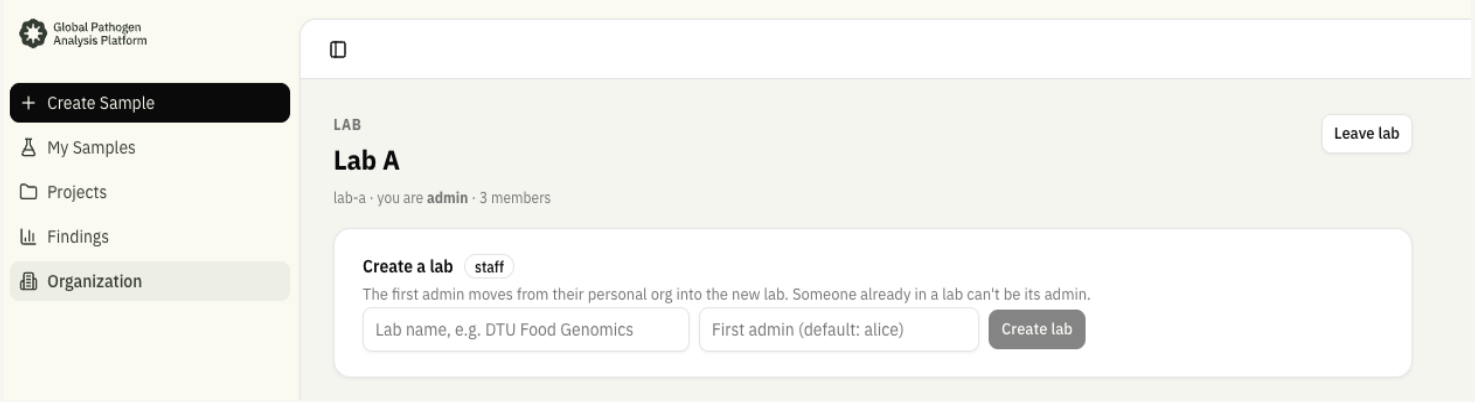
Admin
Can also manage members and clean up

The website, the command line and the assistant all go through the same check.

12 CLIENTS

The same platform works through every client

The interface is only one view of the system — the same secured endpoints also power terminal tools, scripts and AI agents.



OTHER CLIENTS USE THE SAME CONTRACT

curl / terminal

```
curl …/samples
```

Python scripts

```
client.samples.list()
```

AI agents

```
tool: list_samples
```

ONE SECURED API

`/samples` `/files`
`/analyses` `/runs`
`/results` `/projects`

Same auth · validation · behavior

Build the capability once; web, CLI, scripts and agents all consume it.

13 FUTURE

The next phase is about trust, scale and easier use

Five areas to harden as the platform moves from architecture to sustained operation.

1 **Harden auth**

Tighten identity, authorization rules, sessions and administrative controls.

2 **Natural metadata**

Make ingestion and correction intuitive, guided and resilient to imperfect inputs.

3 **Scale scheduling**

Improve compute queues, prioritization, retries and capacity-aware execution.

4 **AI integration**

Use agents to help discover data, configure runs and summarize results safely.

5 **Security and assurance**

Strengthen isolation, auditability, provenance and operational monitoring.

QUESTIONS?